Definitions of Managed Objects
for the DS0 and DS0 Bundle Interface Type


draft-ietf-trunkmib-ds0-mib-07.txt

David Fowler (editor)
Newbridge Networks
davef@newbridge.com

Status of this Memo

This document is an Internet-Draft. Internet-Drafts are working
documents of the Internet Engineering Task Force (IETF), its areas,
and its working groups. Note that other groups may also distribute
working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months
and may be updated, replaced, or obsoleted by other documents at any
time. It is inappropriate to use Internet-Drafts as reference
material or to cite them other than as "work in progress."

To learn the current status of any Internet-Draft, please check the
"1id-abstracts.txt" listing contained in the Internet-Drafts Shadow
Directories on ftp.ietf.org (US East Coast), nic.nordu.net
(Europe), ftp.isi.edu (US West Coast), or munnari.oz.au (Pacific
Rim).

Abstract

This memo defines an experimental portion of the Management
Information Base (MIB) for use with network management protocols in
the Internet community. In particular, it describes objects used for
managing DS0 and DS0 Bundle interfaces. This document is a companion
document with Definitions of Managed Objects for the DS1/E1/DS2/E2,
DS3/E3 and SONET/SDH Interface Types, RFC XXXX [6], RFC XXXX [7] and
RFC XXXX [8].

This memo specifies a MIB module in a manner that is both compliant
to the SNMPv2 SMI, and semantically identical to the peer SNMPv1
definitions.
This memo does not specify a standard for the Internet community.

1. The SNMPv2 Network Management Framework

The SNMPv2 Network Management Framework consists of three major components. They are:

- the SMI, described in RFC 1902 [1] - the mechanisms used for describing and naming objects for the purpose of management.


- the protocol, STD 15, RFC 1157 [3] and/or RFC 1905 [4], - the protocol for accessing managed objects.

The Framework permits new objects to be defined for the purpose of experimentation and evaluation.
2. Objects

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are defined using the subset of Abstract Syntax Notation One (ASN.1) defined in the SMI. In particular, each object type is named by an OBJECT IDENTIFIER, an administratively assigned name. The object type together with an object instance serves to uniquely identify a specific instantiation of the object. For human convenience, we often use a textual string, termed the descriptor, to also refer to the object type.
3. Overview

These objects are used when the particular media being used to realize an interface is a DS0 interface. At present, this applies to these values of the ifType variable in the Internet-standard MIB:

- ds0 (81)
- ds0Bundle (82)

3.1. BONDing Terminology

Please reference The BONDing Spec [9] for definitions of terms used to describe bonding modes.

3.2. Use of ifTable for DS0 Layer

The following items are defined in RFC 2233 [5]. Only the ifGeneralInformationGroup and ifCounterDiscontinuityGroup need to be supported.

<table>
<thead>
<tr>
<th>ifTable Object</th>
<th>Use for DS0 Layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>ifIndex</td>
<td>Interface index.</td>
</tr>
<tr>
<td>ifDescr</td>
<td>See interfaces MIB [5].</td>
</tr>
<tr>
<td>ifType</td>
<td>ds0(81) or ds0Bundle(82).</td>
</tr>
<tr>
<td>ifSpeed</td>
<td>64000 for ds0 (regardless of the setting of robbed bit signalling) or N*64000 for ds0Bundle.</td>
</tr>
<tr>
<td>ifPhysAddress</td>
<td>The value of the Circuit Identifier. If no Circuit Identifier has been assigned this object should have an octet string with zero length.</td>
</tr>
<tr>
<td>ifAdminStatus</td>
<td>See interfaces MIB [5].</td>
</tr>
<tr>
<td>ifOperStatus</td>
<td>See interfaces MIB [5].</td>
</tr>
</tbody>
</table>
ifLastChange See interfaces MIB [5].
ifName See interfaces MIB [5].
ifLinkUpDownTrapEnable Set to disabled(2). Supports read-only access.
ifHighSpeed Set to rounded ifSpeed/1000000.
ifConnectorPresent Set to false(2).

3.3. Using ifStackTable

This section describes by example how to use ifStackTable to represent the relationship of ds0 and ds0Bundles with ds1 interfaces. Implementors of the stack table for ds0 and ds0Bundle interfaces should look at the appropriate RFC for the service being stacked on ds0s and ds0Bundles. Examples given below are for illustration purposes only.

Example: A Frame Relay Service is being carried on 4 ds0s of a ds1.

```
+---------------------+
| Frame Relay Service |
+---------------------+
+---------------------+
| ds0Bundle           |
+---------------------+
|                     |
|                     |
| ds0  | ds0  | ds0  | ds0  |
| +-----+ +-----+ +-----+ +-----+ |
|      |      |      |      |
| ds0  | ds0  | ds0  | ds0  |
+-----+ +-----+ +-----+ +-----+ |
|     |     |     |     |
| ds1  |
+---------------------+
```

The assignment of the index values could for example be:

```
ifIndex Description
1   FrameRelayService (type 44)
2   ds0Bundle (type 82)
3   ds0 #1 (type 81)
```
<table>
<thead>
<tr>
<th>ds0 #2</th>
<th>(type 81)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ds0 #3</td>
<td>(type 81)</td>
</tr>
<tr>
<td>ds0 #4</td>
<td>(type 81)</td>
</tr>
<tr>
<td>dsl</td>
<td>(type 18)</td>
</tr>
</tbody>
</table>
The ifStackTable is then used to show the relationships between the various interfaces.

ifStackTable Entries

<table>
<thead>
<tr>
<th>HigherLayer</th>
<th>LowerLayer</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
</tr>
</tbody>
</table>

In the case where the frameRelayService is using a single ds0, then the ds0Bundle is not required.

```
+---------------------+
| Frame Relay Service |
+---------------------+

++---+
|ds0|
++---+
```

The assignment of the index values could for example be:

<table>
<thead>
<tr>
<th>ifIndex</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>FrameRelayService (type 44)</td>
</tr>
<tr>
<td>2</td>
<td>ds0               (type 81)</td>
</tr>
<tr>
<td>3</td>
<td>ds1               (type 18)</td>
</tr>
</tbody>
</table>
The ifStackTable is then used to show the relationships between the various interfaces.

\[
\begin{array}{c|c}
\text{HigherLayer} & \text{LowerLayer} \\
0 & 1 \\
1 & 2 \\
2 & 3 \\
3 & 0 \\
\end{array}
\]

3.3.1. Usage of Channelization for DS3, DS1, DS0

An example is given here to explain the channelization objects in the DS3, DS1, and DS0 MIBs to help the implementor use the objects correctly. Treatment of E3 and E1 would be similar, with the number of DS0s being different depending on the framing of the E1. Timeslot 16 is not created for framing types that do not pass data over it.

Assume that a DS3 (with ifIndex 1) is channelized into DS1s (without DS2s). The object dsx3Channelization is set to enabledDs1. There will be 28 DS1s in the ifTable. Assume the entries in the ifTable for the DS1s are created in channel order and the ifIndex values are 2 through 29. In the DS1 MIB, there will be an entry in the dsx1ChanMappingTable for each ds1. The entries will be as follows:

\[
\begin{array}{c|c|c}
\text{dsx1ChanMappingTable Entries} & \\
\text{ifIndex} & \text{dsx1Ds1ChannelNumber} & \text{dsx1ChanMappedIfIndex} \\
1 & 1 & 2 \\
1 & 2 & 3 \\
...... & & \\
1 & 28 & 29 \\
\end{array}
\]

In addition, the DS1s are channelized into DS0s. The object dsx1Channelization is set to enabledDs0 for each DS1. When this object is set to this value, 24 DS0s are created by the agent. There will be 24 DS0s in the ifTable for each DS1. If the dsx1Channelization is set to disabled, the 24 DS0s are destroyed.

Assume the entries in the ifTable are created in channel order and the ifIndex values for the DS0s in the first DS1 are 30 through 53. In the DS0 MIB, there will be an entry in the dsx0ChanMappingTable for each DS0. The entries will be as follows:
3.3.2. Usage of ifIndex Mapping for DS0Bundle

An example is given here to explain the ifIndex mapping objects in the DS0Bundle MIB to help the implementor use the objects correctly.

Assume that a DS1 (with ifIndex 1) is channelized into DS0s. There will be 24 DS0s in the ifTable. Assume the entries in the ifTable for the DS0s are created in channel order and the ifIndex values are 2 through 25. Now, assume that there are two bundles on the DS1. The first one uses channels 1 and 2. The second uses channels 3 and 4. There will be two ifTable entries for these bundles, with values of 26 and 27 for ifIndex. There will be an entry in the dsx0BundleTable for each bundle. The entries will be as follows:

dsx0BundleTable Entries

<table>
<thead>
<tr>
<th>dsx0BundleIndex</th>
<th>dsx0BundleIfIndex</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>26</td>
</tr>
<tr>
<td>2</td>
<td>27</td>
</tr>
</tbody>
</table>

There will be an entry in the dsx0ConfigTable for each DS0. The entries will be as follows:

dsx0ConfigTable Entries

<table>
<thead>
<tr>
<th>ifIndex</th>
<th>dsx0Ds0ChannelNumber</th>
<th>dsx0Ds0BundleMappedIfIndex</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1</td>
<td>26</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>26</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>27</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>27</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>......</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>24</td>
<td>0</td>
</tr>
</tbody>
</table>
4. Overview of the MIB

This document contains 2 MIB modules, the DS0 MIB and the DS0Bundle MIB.

4.1. DS0 MIB

The DS0 MIB is used to represent individual DS0s in a DS1 or E1. Variables in this MIB would be created for each DS0 in the ifTable. This MIB contains the following group:

The DS0 Config Group - This group contains configuration information about a particular DS0.

4.2. DS0Bundle MIB

The DS0Bundle MIB is used to represent collections of DS0s that are used together to carry data within a DS1/E1 at speeds greater than that of a single DS0. DS0Bundles are created on top of DS0s and are represented that way in the ifStackTable. This MIB contains the following groups:

The DS0 Bundle Group - This group contains objects used for creating new ds0Bundles. This group is mandatory.

The DS0 Bonding Group - This group contains information about bonding for a ds0Bundle, if bonding is enabled. This group is optional.
5. Object Definitions for DS0

DS0-MIB DEFINITIONS ::= BEGIN

IMPORTS
MODULE-IDENTITY, OBJECT-TYPE,
transmission FROM SNMPv2-SMI
MODULE-COMPLIANCE, OBJECT-GROUP FROM SNMPv2-CONF
DisplayString, TruthValue FROM SNMPv2-TC
ifIndex, InterfaceIndex,
InterfaceIndexOrZero FROM IF-MIB;

-- This is the MIB module for the DS0 Interface objects.

ds0 MODULE-IDENTITY
LAST-UPDATED "9807161630Z"
ORGANIZATION "IETF Trunk MIB Working Group"
CONTACT-INFO
    "        David Fowler
    Postal: Newbridge Networks Corporation
            600 March Road
            Kanata, Ontario, Canada K2K 2E6
            Tel: +1 613 591 3600
            Fax: +1 613 599 3619
            E-mail: davef@newbridge.com"
DESCRIPTION
    "The MIB module to describe
DS0 interfaces objects."
REVISION "9805242010Z"
DESCRIPTION
    "Initial version of the DS0-MIB."
::= { transmission 81 }
-- The DS0 Config Group

-- Implementation of this group is mandatory for all systems that use a DS0 Interface.

-- The DS0 Config Group consists of two tables:
--   DS0 Configuration Table
--   DS0 Channel Mapping Table

-- The DS0 Configuration Table

ds0ConfigTable OBJECT-TYPE
SYNTAX  SEQUENCE OF Dsx0ConfigEntry
MAX-ACCESS  not-accessible
STATUS  current
DESCRIPTION
   "The DS0 Configuration table."
::= { ds0 1 }

Dsx0ConfigEntry OBJECT-TYPE
SYNTAX  Dsx0ConfigEntry
MAX-ACCESS  not-accessible
STATUS  current
DESCRIPTION
   "An entry in the DS0 Configuration table. There is an entry in this table for each DS0 interface."
INDEX   { ifIndex }
::= { ds0ConfigTable 1 }

Ds0ConfigEntry ::= 
SEQUENCE {
   ds0Ds0ChannelNumber        INTEGER,
   ds0RobbedBitSignalling     TruthValue,
   ds0CircuitIdentifier       DisplayString,
   ds0IdleCode                INTEGER,
   ds0SeizedCode              INTEGER,
   ds0ReceivedCode            INTEGER,
   ds0TransmitCodesEnable     TruthValue,
   ds0Ds0BundleMappedIfIndex  InterfaceIndexOrZero
}

dsx0Ds0ChannelNumber OBJECT-TYPE
SYNTAX  INTEGER(0..31)
MAX-ACCESS read-only
STATUS  current

Expires January 1999
DESCRIPTION
"This object indicates the channel number of the
ds0 on its DS1/E1."
::= { dsx0ConfigEntry 1 }

dsx0RobbedBitSignalling OBJECT-TYPE
SYNTAX  TruthValue
MAX-ACCESS  read-write
STATUS  current
DESCRIPTION
"This object indicates if Robbed Bit Signalling is
turned on or off for a given ds0. This only
applies to DS0s on a DS1 link. For E1 links the
value is always off (false)."
::= { dsx0ConfigEntry 2 }

dsx0CircuitIdentifier OBJECT-TYPE
SYNTAX  DisplayString (SIZE (0..255))
MAX-ACCESS  read-write
STATUS  current
DESCRIPTION
"This object contains the transmission vendor’s
circuit identifier, for the purpose of
facilitating troubleshooting."
::= { dsx0ConfigEntry 3 }

dsx0IdleCode OBJECT-TYPE
SYNTAX  INTEGER(0..15)
MAX-ACCESS  read-write
STATUS  current
DESCRIPTION
"This object contains the code transmitted in the
ABCD bits when the ds0 is not connected and
dsx0TransmitCodesEnable is enabled. The object is
a bitmap and the various bit positions are:
1   D bit
2   C bit
4   B bit
8   A bit"
::= { dsx0ConfigEntry 4 }

dsx0SeizedCode OBJECT-TYPE
SYNTAX  INTEGER(0..15)
MAX-ACCESS  read-write
STATUS  current
DESCRIPTION
"This object contains the code transmitted in the
ABCD bits when the ds0 is connected and
dsx0TransmitCodesEnable is enabled. The object is
a bitmap and the various bit positions are:
  1  D bit
  2  C bit
  4  B bit
  8  A bit"
::= { dsx0ConfigEntry 5 }

dsx0ReceivedCode OBJECT-TYPE
SYNTAX INTEGER(0..15)
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object contains the code being received in
the ABCD bits. The object is a bitmap and the
various bit positions are:
  1  D bit
  2  C bit
  4  B bit
  8  A bit"
::= { dsx0ConfigEntry 6 }

dsx0TransmitCodesEnable OBJECT-TYPE
SYNTAX TruthValue
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"This object determines if the idle and seized
codes are transmitted. If the value of this object
is true then the codes are transmitted."
::= { dsx0ConfigEntry 7 }

dsx0Ds0BundleMappedIfIndex OBJECT-TYPE
SYNTAX InterfaceIndexOrZero
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object indicates the ifIndex value assigned
by the agent for the ds0Bundle(82) ifEntry to
which the given ds0(81) ifEntry may belong.

If the given ds0(81) ifEntry does not belong to
any ds0Bundle(82) ifEntry, then this object has a
value of zero.

While this object provides information that can
also be found in the ifStackTable, it provides
this same information with a single table lookup,
rather than by walking the ifStackTable to find
the possibly non-existent ds0Bundle(82) ifEntry
that may be stacked above the given ds0(81)
ifTable entry."

::= { dsx0ConfigEntry 8 }

-- The DS0 Channel Mapping Table

dsx0ChanMappingTable OBJECT-TYPE
SYNTAX  SEQUENCE OF Dsx0ChanMappingEntry
MAX-ACCESS not-accessible
STATUS  current
DESCRIPTION
"The DS0 Channel Mapping table.  This table maps a
DS0 channel number on a particular DS1/E1 into an
ifIndex."

::= { ds0 3 }

dsx0ChanMappingEntry OBJECT-TYPE
SYNTAX  Dsx0ChanMappingEntry
MAX-ACCESS not-accessible
STATUS  current
DESCRIPTION
"An entry in the DS0 Channel Mapping table.  There
is an entry in this table corresponding to each
ds0 ifEntry within any interface that is
channelized to the individual ds0 ifEntry level.

This table is intended to facilitate mapping from
channelized interface / channel number to DS0
ifEntry.  (e.g. mapping (DS1 ifIndex, DS0 Channel
Number) -> ifIndex)

While this table provides information that can
also be found in the ifStackTable and
dsx0ConfigTable, it provides this same information
with a single table lookup, rather than by walking
the ifStackTable to find the various constituent
ds0 ifTable entries, and testing various
dsx0ConfigTable entries to check for the entry
with the applicable DS0 channel number.
INDEX  { ifIndex, dsx0Ds0ChannelNumber }
::= { dsx0ChanMappingTable 1 }

Dsx0ChanMappingEntry ::= SEQUENCE {
    dsx0ChanMappedIfIndex  InterfaceIndex
}

dsx0ChanMappedIfIndex OBJECT-TYPE
SYNTAX  InterfaceIndex
MAX-ACCESS  read-only
STATUS  current
DESCRIPTION
"This object indicates the ifIndex value assigned
by the agent for the individual ds0 ifEntry that
 corresponds to the given DS0 channel number
(specified by the INDEX element
dsx0Ds0ChannelNumber) of the given channelized
interface (specified by INDEX element ifIndex)."
::= { dsx0ChanMappingEntry 1 }

-- conformance information

ds0Conformance OBJECT IDENTIFIER ::= { ds0 2 }
ds0Groups OBJECT IDENTIFIER ::= { ds0Conformance 1 }
ds0Compliances OBJECT IDENTIFIER ::= { ds0Conformance 2 }

-- compliance statements

ds0Compliance MODULE-COMPLIANCE
STATUS  current
DESCRIPTION
"The compliance statement for DS0 interfaces."
MODULE  -- this module
    MANDATORY-GROUPS { ds0ConfigGroup }

OBJECT dsx0RobbedBitSignalling
MIN-ACCESS read-only
DESCRIPTION
"The ability to set RBS is not required."
OBJECT dsx0CircuitIdentifier
MIN-ACCESS read-only
DESCRIPTION
"The ability to set the circuit identifier is not required."

OBJECT dsx0IdleCode
MIN-ACCESS read-only
DESCRIPTION
"The ability to set the idle code is not required."

OBJECT dsx0SeizedCode
MIN-ACCESS read-only
DESCRIPTION
"The ability to set the seized code is not required."

OBJECT dsx0TransmitCodesEnable
MIN-ACCESS read-only
DESCRIPTION
"The ability to enable and disable the transmitting of idle and seized codes is not required."

::= { ds0Compliances 1 }

-- units of conformance

ds0ConfigGroup OBJECT-GROUP
OBJECTS { dsx0Ds0ChannelNumber,
    dsx0RobbedBitSignalling,
    dsx0CircuitIdentifier,
    dsx0IdleCode,
    dsx0SeizedCode,
    dsx0ReceivedCode,
    dsx0TransmitCodesEnable,
    dsx0Ds0BundleMappedIfIndex,
    dsx0ChanMappedIfIndex }

STATUS current
DESCRIPTION
"A collection of objects providing configuration information applicable to all DS0 interfaces."

::= { ds0Groups 1 }
6. Object Definitions for DS0 Bundle

DS0BUNDLE-MIB DEFINITIONS ::= BEGIN

IMPORTS
    MODULE-IDENTITY, OBJECT-TYPE, transmission
    FROM SNMPv2-SMI
    MODULE-COMPLIANCE, OBJECT-GROUP
    FROM SNMPv2-CONF
    DisplayString, RowStatus,
    TestAndIncr
    FROM SNMPv2-TC
    ifIndex, InterfaceIndex
    FROM IF-MIB;

-- This is the MIB module for the DS0Bundle Interface
-- objects.

ds0Bundle MODULE-IDENTITY
    LAST-UPDATED "9807161630Z"
    ORGANIZATION "IETF Trunk MIB Working Group"
    CONTACT-INFO
        "        David Fowler
        Postal: Newbridge Networks Corporation
        600 March Road
        Kanata, Ontario, Canada K2K 2E6

        Tel: +1 613 591 3600
        Fax: +1 613 599 3619

        E-mail: davef@newbridge.com"
    DESCRIPTION
        "The MIB module to describe
         DS0 Bundle interfaces objects."
    REVISION "9805242010Z"
    DESCRIPTION
        "Initial version of the DS0BUNDLE-MIB."

 ::= { transmission 82 }
-- The DS0 Bundle Config Group
-- Implementation of this group is mandatory for all
-- systems that use a DS0Bundle Interface.
-- The DS0 Bundle Config Group consists of one table:
--    DS0 Bundle Table
-- The DS0 Bundle Table

dsx0BundleNextIndex OBJECT-TYPE
SYNTAX  TestAndIncr
MAX-ACCESS  read-write
STATUS  current
DESCRIPTION
"This object is used to assist the manager in
selecting a value for dsx0BundleIndex. Because
this object is of syntax TestAndIncr (see the
SNMPv2-TC document, RFC 1903) it can also be used
to avoid race conditions with multiple managers
trying to create rows in the table.

If the result of the SET for dsx0BundleNextIndex
is not success, this means the value has been
changed from index (i.e. another manager used the
value), so a new value is required.

The algorithm is:
done = false
while done == false
    index = GET (dsx0BundleNextIndex.0)
    SET (dsx0BundleNextIndex.0=index)
    if (set failed)
        done = false
    else
        SET(dsx0BundleRowStatus.index=createAndGo)
        if (set failed)
            done = false
        else
            done = true
other error handling"
 ::= { ds0Bundle 2 }

dsx0BundleTable OBJECT-TYPE
SYNTAX  SEQUENCE OF Dsx0BundleEntry
MAX-ACCESS  not-accessible
STATUS  current
DESCRIPTION
  "There is an row in this table for each ds0Bundle
in the system. This table can be used to
(indirectly) create rows in the ifTable with
ifType = 'ds0Bundle(82)'.'"
 ::= { ds0Bundle 3 }

Dsx0BundleEntry OBJECT-TYPE
SYNTAX  Dsx0BundleEntry
MAX-ACCESS  not-accessible
STATUS  current
DESCRIPTION
  "There is a row in entry in this table for each
ds0Bundle interface."
INDEX  { dsx0BundleIndex }
 ::= { dsx0BundleTable 1 }

Dsx0BundleEntry ::= SEQUENCE {
    dsx0BundleIndex              INTEGER,
    dsx0BundleIfIndex            InterfaceIndex,
    dsx0BundleCircuitIdentifier  DisplayString,
    dsx0BundleRowStatus          RowStatus
}

dsx0BundleIndex OBJECT-TYPE
SYNTAX  INTEGER (0..2147483647)
MAX-ACCESS  not-accessible
STATUS  current
DESCRIPTION
  "A unique identifier for a ds0Bundle. This is not
the same value as ifIndex. This table is not
indexed by ifIndex because the manager has to
choose the index in a createable row and the agent
must be allowed to select ifIndex values."
 ::= { dsx0BundleEntry 1 }

dsx0BundleIfIndex OBJECT-TYPE
SYNTAX  InterfaceIndex
MAX-ACCESS  read-only
STATUS  current
DESCRIPTION
"The ifIndex value the agent selected for the (new) ds0Bundle interface."
::= { ds0BundleEntry 2 }

dsx0BundleCircuitIdentifier OBJECT-TYPE
SYNTAX  DisplayString (SIZE (0..255))
MAX-ACCESS  read-create
STATUS  current
DESCRIPTION
"This variable contains the transmission vendor’s circuit identifier, for the purpose of facilitating troubleshooting."
::= { ds0BundleEntry 3 }

dsx0BundleRowStatus OBJECT-TYPE
SYNTAX  RowStatus
MAX-ACCESS  read-create
STATUS  current
DESCRIPTION
"This object is used to create and delete rows in this table."
::= { ds0BundleEntry 4 }

-- The DS0 Bonding Group

-- Implementation of this group is optional for all systems that use a DS0Bundle Interface.

-- The DS0 Bonding Group consists of one table:
--  DS0 Bonding Table

-- The DS0 Bonding Table

dsx0BondingTable OBJECT-TYPE
SYNTAX  SEQUENCE OF Dsx0BondingEntry
MAX-ACCESS  not-accessible
STATUS  current
DESCRIPTION
"The DS0 Bonding table."
::= { ds0Bundle 1 }

dsx0BondingEntry OBJECT-TYPE
SYNTAX  Dsx0BondingEntry
MAX-ACCESS  not-accessible
STATUS  current
DESCRIPTION
"An entry in the DS0 Bonding table. There is a row in this table for each DS0Bundle interface."

INDEX  { ifIndex }
 ::=  { dsx0BondingTable 1 }

Dsx0BondingEntry ::=  
 SEQUENCE {
   dsx0BondMode            INTEGER,  
   dsx0BondStatus          INTEGER,  
   dsx0BondRowStatus       RowStatus

   }

Dsx0BondMode OBJECT-TYPE
SYNTAX  INTEGER {
    none(1),
    other(2),
    mode0(3),
    mode1(4),
    mode2(5),
    mode3(6)
}

MAX-ACCESS  read-create
STATUS  current
DESCRIPTION
"This object indicates which BONDing mode is used, if any, for a ds0Bundle. Mode0 provides parameter and number exchange with no synchronization. Mode 1 provides parameter and number exchange. Mode 1 also provides synchronization during initialization but does not include inband monitoring. Mode 2 provides all of the above plus inband monitoring. Mode 2 also steals 1/64th of the bandwidth of each channel (thus not supporting n x 56/64 kbit/s data channels for most values of n). Mode 3 provides all of the above, but also provides n x 56/64 kbit/s data channels. Most common implementations of Mode 3 add an extra channel to support the inband monitoring overhead. ModeNone should be used when the interface is not performing bandwidth-on-demand."

 ::=  { dsx0BondingEntry 1 }

dsx0BondStatus OBJECT-TYPE
SYNTAX  INTEGER {

idle(1),
callSetup(2),
dataTransfer(3)
}
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object indicates the current status of the bonding call using this ds0Bundle. idle(1) should be used when the bonding mode is set to none(1)."
 ::= { dsx0BondingEntry 2 }

dsx0BondRowStatus OBJECT-TYPE
SYNTAX  RowStatus
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"This object is used to create new rows in this table, modify existing rows, and to delete existing rows."
 ::= { dsx0BondingEntry 3 }

-- conformance information

ds0BundleConformance OBJECT IDENTIFIER ::= { ds0Bundle 4 }

ds0BundleGroups OBJECT IDENTIFIER ::= { ds0BundleConformance 1 }

ds0BundleCompliances OBJECT IDENTIFIER ::= { ds0BundleConformance 2 }

-- compliance statements

ds0BundleCompliance MODULE-COMPLIANCE
STATUS current
DESCRIPTION
"The compliance statement for DS0Bundle interfaces."
MODULE -- this module
MANDATORY-GROUPS {ds0BundleConfigGroup }

GROUP ds0BondingGroup
DESCRIPTION
 "Implementation of this group is optional for all systems that attach to a DS0Bundle Interface."

OBJECT dsx0BundleRowStatus
SYNTAX INTEGER {
    active(1),
    createAndGo(4),
    destroy(6)
}
MIN-ACCESS read-only
DESCRIPTION
 "The agent is not required to support a SET operation to this object, and only three of the six enumerated values for the RowStatus textual convention need be supported. Only supporting createAndGo for a creation process prevents the manager from creating an inactive row in the ds0BundleTable. Inactive rows in the ds0BundleTable do not make sense."

OBJECT dsx0BundleCircuitIdentifier
MIN-ACCESS read-only
DESCRIPTION
 "The agent is not required to support a SET operation to this object."

::= { ds0BundleCompliances 1 }

-- units of conformance

ds0BondingGroup OBJECT-GROUP
OBJECTS { dsx0BondMode,
    dsx0BondStatus,
    dsx0BondRowStatus }
STATUS current
DESCRIPTION
 "A collection of objects providing configuration information applicable to all DS0 interfaces."

::= { ds0BundleGroups 1 }
ds0BundleConfigGroup  OBJECT-GROUP
   OBJECTS { dsx0BundleNextIndex,
       dsx0BundleIfIndex,
       dsx0BundleCircuitIdentifier,
       dsx0BundleRowStatus }  
   STATUS  current
   DESCRIPTION
       "A collection of objects providing the ability to
       create a new ds0Bundle in the ifTable as well as
       configuration information about the ds0Bundle."
   ::= { ds0BundleGroups 2  }
END

7. Acknowledgments

This document was produced by the Trunk MIB Working Group:
8. References


9. Security Considerations

This MIB adds no security issues that are not raised by SNMP itself. As a Network Management Protocol, SNMP is capable of changing the configuration of a system, of denial of service attacks through requiring the agent to respond to its messages, and displaying values of system counters and configuration settings in the clear where an adversary might read them. This, however, is not in itself a threat: this is the function of the protocol, but could be misused. Such misuse can be avoided or minimized through SNMP Security.

Setting the following objects to an inappropriate value can cause loss of traffic. In the case of dsx0RobbedBitSignalling, for example, the nature of the traffic flowing on the DS0 can be affected.

- dsx0RobbedBitSignalling
- dsx0IdleCode
- dsx0SeizedCode
- dsx0TransmitCodesEnable
- dsx0BundleRowStatus
- dsx0BondMode
- dsx0BondRowStatus

Setting the following objects is mischievous, but not harmful to traffic.

- dsx0CircuitIdentifier
- dsx0BundleNextIndex

10. Authors’ Addresses

David Fowler
Newbridge Networks
600 March Road
Kanata, Ontario, Canada K2K 2E6

Phone: (613) 599-3600, ext 6559
EMail: davef@newbridge.com
Table of Contents

1 The SNMPv2 Network Management Framework ...................... 2
2 Objects ..................................................................... 3
3 Overview ..................................................................... 4
3.1 BONDing Terminology ................................................. 4
3.2 Use of ifTable for DS0 Layer ........................................ 4
3.3 Using ifStackTable ..................................................... 5
3.3.1 Usage of Channelization for DS3, DS1, DS0 .................. 8
3.3.2 Usage of ifIndex Mapping for DS0Bundle .................... 9
4 Overview of the MIB .................................................. 10
4.1 DS0 MIB .................................................................... 10
4.2 DS0Bundle MIB ......................................................... 10
5 Object Definitions for DS0 .............................................. 11
5.1 The DS0 Config Group ................................................. 12
5.1.1 The DS0 Configuration Table ..................................... 12
5.1.2 The DS0 Channel Mapping Table ................................. 15
6 Object Definitions for DS0 Bundle .................................. 19
6.1 The DS0 Bundle Config Group ..................................... 20
6.1.1 The DS0 Bundle Table ............................................. 20
6.2 The DS0 Bonding Group .............................................. 22
6.2.1 The DS0 Bonding Table .......................................... 22
7 Acknowledgments ...................................................... 26
8 References ............................................................... 27
9 Security Considerations ................................................. 28
10 Authors’ Addresses ..................................................... 28